Knapsack_1

มานีอยากทำอาหารให้มานะทาน จึงเดินทางไปจ่ายตลาดด้วยตัวเอง และด้วยความที่เป็นผู้หญิงจึงต้องการเลือกวัตถุดิบที่ให้คุณค่าทางโภชนาการสูงสุดและมีความหลากหลาย โดยที่น้ำหนักรวมของวัตุดิบต้องไม่เกินกำลังที่มานีจะถือกลับได้

จงเขียนโปรแกรมเพื่อช่วยมานีซื้อวัตถุดิบ

Input: บรรทัดที่ 1 คือ จำนวนชนิดของวัตถุดิบในตลาด (n) และน้ำหนักรวมที่มานีสามารถถือได้ (W)
บรรทัดที่ 2 คือ น้ำหนักของวัตถุดิบแต่ละชนิด (w_1 w_2 w_3 ...w_n)
บรรทัดที่ 3 คือ คุณค่าทางโภชนาการของวัตถุดิบแต่ละชนิด (v_1 v_2 v_3 ...v_n)

Output: คุณค่าทางโภชนาการรวมสูงสุด จำนวนชนิดของวัตถุดิบที่เลือก และน้ำหนักรวมที่มานีต้องถือของกลับ

Example:

Input	Output
4 10 6 3 4 2 30 14 16 9	46 2 10
5 60 20 10 40 10 30 40 100 50 60 30	210 3 60

1 2 3 4

mChain_1

จงเขียนโปรแกรมเพื่อหาจำนวนการคูณกันน้อยที่สุดของ matrix chain ที่กำหนดให้

Input: บรรทัดที่ 1 คือ จำนวนเมทริกซ์ n

บรรทัดที่ 2 คือ ขนาดของเมทริกซ์ p_0 p_1 p_2 p_3 ... p_n (เว้นวรรคข้อมูลแต่ละตัว)

Output: จำนวนการคูณที่น้อยที่สุด

Example:

Input	Output
6 30 20 10 5 5 10 40	12250
3 10 100 5 50	7500

Minimum Cost of Shogun

เนื่องจากสถานการณ์ COVID-19 ทำให้การขายสินค้าออนไลน์ขยายตัวมากขึ้น สินค้าทางการเกษตรอย่างส้มโชกุนก็เช่นกัน สวนส้มโชกุนรูสะมิแลได้ปรับกลยุทธ์มาขายส้มทางออนไลน์โดยจัดขายเป็นลัง ๆ ละตั้งแต่ 1, 2, 3, ..., n กิโลกรัม ราคาแต่ละลังก็ขึ้นอยู่กับคุณภาพของส้มโปรโมชั่น และการจัดส่งในช่วงนั้น ๆ

มานะต้องการสั่งซื้อส้มโชกุนจากสวนดังกล่าวจำนวน w กิโลกรัม และต้องการส้มโชกุนในราคาถูกที่สุด จงเขียนโปรแกรมเพื่อช่วยมานะคำนวณราคาส้มที่ต้องจ่าย

Input: บรรทัดที่ 1 จำนวนส้มโชกุนที่มานะต้องการซื้อ (w) และขนาดลังสูงสุดที่สวนส้มมี (n) โดยที่ทั้งคู่เป็นจำนวนเต็มและเว้นวรรคข้อมูลแต่ละตัว

บรรทัดที่ 2 ราคาส้มโชกุนตั้งแต่ 1 กิโลกรัม ถึง n กิโลกรัม โดยเว้นวรรคข้อมูลแต่ละตัว

Output: ราคาส้มโชกุนที่น้อยที่สุดที่มานะต้องจ่ายจากการซื้อส้ม w กิโลกรัม

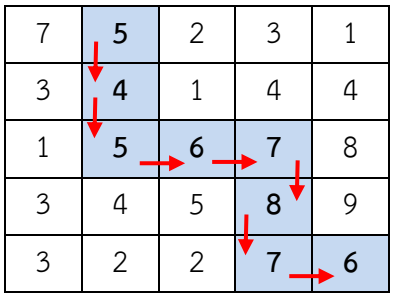
Example:

Input	Output	Note
5 5 20 10 4 50 90	14	มานะซื้อส้ม 2 กก. 1 ลัง และ 3 กก. 1 ลัง
5 5 1 10 4 50 90	5	มานะซื้อส้ม 1 กก. 5 ลัง

Difference-1 Sequence

ลำดับ Difference-1 (Difference-1 Sequence) คือ ลำดับของจำนวนเต็มที่อยู่ในตารางที่มีผลต่างต่างกันเท่ากับหนึ่งเสมอ โดยลำดับสามารถเริ่มที่เซลล์ใด ๆ ก็ได้ แต่เซลล์ในลำดับถัดไปต้องเป็นเซลล์ในตำแหน่งด้านขวา หรือล่างเท่านั้น และลำดับสามารถสิ้นสุดได้ที่เซลล์ใด ๆ เช่นกัน ส่วนความยาวของลำดับ Difference-1 จะนับจากจำนวนการเชื่อมต่อกันของลำดับ

จากตัวอย่าง ลำดับ Difference-1 ที่ยาวที่สุด คือ 5 4 5 6 7 8 7 6 และมีความยาวเท่ากับ 7



จงเขียนโปรแกรมเพื่อหาลำดับ Difference-1 และความยาวที่ยาวที่สุดจากตารางที่กำหนดให้

Input: บรรทัดที่ 1 ขนาดของตาราง (n) โดยที่ $1 \le n \le 1{,}000$
n บรรทัดถัดไป คือ จำนวนเต็ม n ตัว โดยเว้นวรรคข้อมูลแต่ละตัว

Output: บรรทัดที่ 1 ลำดับ Difference-1 ที่ยาวที่สุด โดยเว้นวรรคข้อมูลแต่ละตัว
บรรทัดที่ 2 ความยาวของลำดับ Difference-1 ที่ยาวที่สุด

Example:

Input	Output
5 7 5 2 3 1 3 4 1 4 4 1 5 6 7 8 3 4 5 8 9 3 2 2 7 6	5 4 5 6 7 8 7 6 7

Glass Cutter

กรมทางหลวงได้จ้างช่างตัดหญ้าจำนวน k คน เพื่อตัดหญ้าข้างถนนรูสะมิแล โดยแบ่งช่วงถนนเป็นบล็อก ๆ ตามความยากง่าย และคาดการณ์ไว้ว่าแต่ละบล็อกจะใช้เวลาตัดหญ้านานเท่าใด

จงเขียนโปรแกรมเพื่อคำนวณหาเวลาที่น้อยที่สุดเพื่อตัดหญ้าข้างถนนรูสะมิแล โดยมีข้อกำหนดว่าช่างตัดหญ้าจะเริ่มตัดหญ้าพร้อมกัน และตัดได้เฉพาะบล็อกที่ติดกันเท่านั้น

Input: บรรทัดที่ 1 จำนวนช่างตัดหญ้า (k) และจำนวนบล็อกถนน (n) โดยที่ $1 \le k, n \le 1{,}000{,}000$
บรรทัดที่ 2 คือ เวลาที่คาดการณ์ว่าจะตัดหญ้าแต่ละบล็อกเสร็จ โดยเริ่มจากบล็อกที่ 1, 2, ..., n (เว้นวรรคข้อมูลแต่ละตัว)

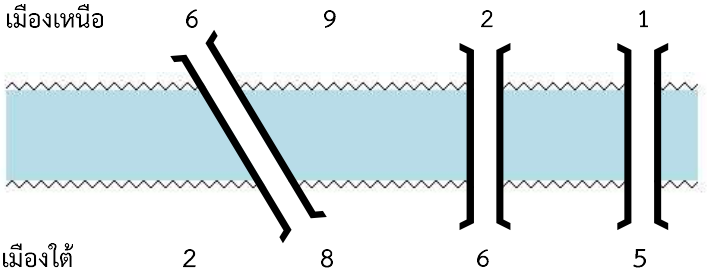
Output: เวลาที่น้อยที่สุดที่ใช้ในการตัดหญ้าจนเสร็จ

Example:

Input	Output	Notes
2 4 10 10 10 10	20	คนที่ 1 ตัดบล็อกที่ 1-2 คนที่ 2 ตัดบล็อกที่ 3-4
2 4 10 20 30 40	60	คนที่ 1 ตัดบล็อกที่ 1-3 คนที่ 2 ตัดบล็อกที่ 4
3 6 10 20 30 50 30 40	70	คนที่ 1 ตัดบล็อกที่ 1-3 คนที่ 2 ตัดบล็อกที่ 4 คนที่ 3 ตัดบล็อกที่ 5-6

Bridges

แม่น้ำรูสะมิแลแบ่งเมืองเป็น 2 ทิศ คือ เมืองทางเหนือและทางใต้ โดยมีจำนวนเมืองเท่ากัน (n) โดยเมืองทางเหนือประกอบด้วยเมือง N_1, N_2, ..., N_n และเมืองทางใต้ประกอบด้วยเมือง S_1, S_2, ..., S_n ซึ่งแต่ละเมืองจะมีหมายเลขประจำเมืองกำกับไว้ (เมืองทางเหนือและทางใต้สามารถมีหมายเลขซ้ำกันได้ และเมืองในฝั่งเดียวกันก็ซ้ำกันได้เช่นกัน) ผู้ปกครองเมืองต้องการสร้างสะพานเพื่อเชื่อมระหว่างเมืองทางเหนือและทางใต้ให้มากที่สุด โดยมีกฎว่าห้ามสร้างสะพานตัดกัน และหมายเลขประจำเมืองทางเหนือต้องมีค่าน้อยกว่าหรือเท่ากับหมายเลขประจำเมืองทางใต้เท่านั้น เช่น จากรูปสามารถสร้างสะพานได้สูงสุด 3 สะพาน และไม่สามารถสร้างสะพานระหว่างเมือง N_2-S_5 ที่ตัดกับ N_1-S_6 ได้



จงเขียนโปรแกรมเพื่อหาจำนวนสะพานสูงสุดที่สามารถสร้างได้ตามเงื่อนไขข้างต้น

Input: บรรทัดที่ 1 จำนวนเมือง (n) เป็นจำนวนเต็ม โดยที่ $1 \le n \le 100,000$

บรรทัดที่ 2 คือ หมายเลขประจำเมืองเหนือ n จำนวน (เว้นวรรคข้อมูลแต่ละตัว)

บรรทัดที่ 3 คือ หมายเลขประจำเมืองใต้ n จำนวน ซึ่งซ้ำกับหมายเลขประจำเมืองเหนือได้ (เว้นวรรคข้อมูลแต่ละตัว)

Output: จำนวนสะพานสูงสุดที่สามารถสร้างได้

Example:

Input	Output
4 6 9 2 1 2 8 6 5	3
4 6 4 2 1 2 3 6 5	2
4 6 4 2 1 6 4 2 1	4

Gold Price

90/100

ณ เมืองรูสะมิแลมีการกำหนดการซื้อขายทองคำแท่งวันละครั้ง โดยกำหนดราคาซื้อและราคาขายเท่ากัน แต่ราคาจะขึ้นลงแตกต่างกันในแต่ละวัน นักเก็งกำไรมักจะซื้อทองคำแท่งในวันที่คิดว่าราคาถูก และขายในวันที่คิดว่าทองราคาแพง มานะเป็นนักเก็งกำไรจากการซื้อขายทองคำแท่งที่เดินทางมาจากเมืองทิพย์ โดยแต่ละครั้งที่เดินทางมานะจะทราบราคาซื้อขายทองคำแท่งล่วงหน้า n วัน เพื่อไม่ให้ชาวเมืองรูสะมิแลสงสัยเรื่องการเก็งกำไรจากการซื้อขายทองของมานะ มานะจึงมีกฎประจำตัวว่าจะทำการซื้อขายทองคราวละหนึ่งแท่งเท่านั้น และทำการซื้อขายไม่เกิน k ครั้ง โดยแต่ละการซื้อขาย มานะจะทำการซื้อและขายให้เสร็จสิ้นก่อนทำการซื้อขายครั้งถัดไป หากการซื้อขายครั้งใดขาดทุน (กำไรเป็นศูนย์หรือติดลบ) มานะจะไม่ทำการซื้อขาย

จงเขียนโปรแกรมเพื่อหาว่าแต่ละครั้งที่มานะเดินทางมาที่เมืองรูสะมิแล มานะจะทำกำไรจากการซื้อขายทองได้สูงสุดเท่าใด

Input: บรรทัดที่ 1 จำนวนครั้งที่จะทำการซื้อขายทอง (k) และจำนวนวันที่มานะรู้ราคาทองล่วงหน้า (n)
โดยที่ $1 \le k \le n/2$ และ $1 \le n \le 1{,}000$
บรรทัดที่ 2 คือ ราคาซื้อขายทองคำแท่งในแต่ละวัน จำนวน n วัน (เป็นจำนวนเต็ม)

Output: กำไรสูงสุดจากการซื้อขายทองคำแท่งของมานะ

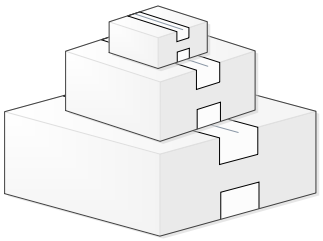
Example:

Input	Output	Note
2 6 100 220 50 750 650 800	870	ครั้งที่ 1: ซื้อที่ราคา 100 และขายที่ 220 -> กำไร 120 ครั้งที่ 2: ซื้อที่ราคา 50 และขายที่ 800 -> กำไร 750
3 8 10 50 20 30 70 60 40 50	100	ครั้งที่ 1: ซื้อที่ราคา 10 และขายที่ 50 -> กำไร 40 ครั้งที่ 2: ซื้อที่ราคา 20 และขายที่ 70 -> กำไร 50 ครั้งที่ 3: ซื้อที่ราคา 40 และขายที่ 50 -> กำไร 10
2 5 1000 900 1200 1100 1000	300	ครั้งที่ 1: ซื้อที่ราคา 900 และขายที่ 1200 -> กำไร 300 ครั้งที่ 2: ไม่ทำการซื้อขาย -> กำไร 0

Stack of Boxes

ปัญหาการซ้อนกล่อง คือ การซ้อนกล่องสี่เหลี่ยมให้ได้ความสูงที่สูงที่สุด โดยมีเงื่อนไขต่อไปนี้

1) กล่องที่อยู่ด้านล่าง ฐานที่วาง (กว้าง x ลึก) ต้องมีฐานขนาดใหญ่กว่ากล่องด้านบน
2) สามารถหมุนกล่องให้กล่องด้านใดเป็นฐานก็ได้
3) สามารถใช้กล่องที่มีขนาดเท่ากันในการจัดเรียงได้ โดยการหมุนกล่องเพื่อเปลี่ยนเอาด้านอื่นวางเป็นฐาน (เพื่อให้ตรงกับเงื่อนไขข้อที่ 1)



จงเขียนโปรแกรมเพื่อหาความสูงที่มากที่สุดที่ได้จากการซ้อนกล่องตามเงื่อนไขกำหนด

Input: บรรทัดที่ 1 คือ จำนวนขนาดของกล่อง n ขนาด
บรรทัดที่ 2 คือ ขนาดของกล่องใบแรก ได้แก่ ความสูง ความกว้าง และความลึก (เว้นวรรคข้อมูลแต่ละตัว)
บรรทัดถัดไป n-1 บรรทัด คือ ขนาดของกล่องที่เหลือ

Output: ความสูงที่มากที่สุดที่ได้จากการซ้อนกล่อง

Sample:

Input	Output	Note
4 4 6 7 1 2 3 4 5 6 10 12 32	45	ซ้อนกล่องจากบนลงล่างได้ดังนี้ (สูง x กว้าง x ลึก) {3, 1, 2}, {6, 4, 5}, {4, 6, 7}, {32, 10, 12}